NPS55-77-25

NAVAL POSTGRADUATE SCHOOL

Monterey, California

APPROXIMATE MODELS FOR CENTRAL SERVER SYSTEMS

WITH TWO JOB TYPES

by

J. P. Lehoczky

and

D. P. Gaver

June 1977

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFONRIA

Rear Admiral Isham Linder
Superintendent

Jack R. Borsting
Provost

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NPS55-77-25	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* Approximate Models for Central Server Systems with Two Job Types		5. TYPE OF REPORT & PERIOD COVERED Technical Report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR*(s)* D. P. Gaver and J. P. Lehoczky		8. CONTRACT OR GRANT NUMBER*(s)*
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, Ca. 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, Ca. 93940		12. REPORT DATE June 1977 13. NUMBER OF PAGES 26
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Performance evaluation; Closed queueing network; State space reduction; Phase-type distributions, Approximate model.

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

Two new approximation techniques for describing the performance of a closed queueing network when two or more job types are present are introduced and analyzed. The approximations apply in cases where a product form steady state solution can not be obtained, for example, in the first-come-first-served single server queue case. This approximation provides a method of reducing the state space to a small fraction of that needed for an implementation of the Gaver-Humfeld method. Numerical results illustrate the excellent accuracy of these techniques.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

APPROXIMATE MODELS FOR CENTRAL SERVER SYSTEMS WITH TWO JOB TYPES

by

J. P. Lehoczky*

D. P. Gaver**

1. INTRODUCTION

1.1 General Summary

The purpose of this paper is to provide a new approximation technique for describing the performance of a closed queueing network when there are two or more job types present, and the order of service (scheduling policy) of interest does not make system states Markovian in the number present at each server; "first-come, first-served" order is an example. The model also allows incorporation of non-exponential distributions to represent service time distributions. Our technique drastically reduces the size of the state space used to describe complex processes. Consequently, it facilitates calculation of state probabilities and system performance measures through use of numerical methods such as Gauss-Seidel iteration. It is also helpful in simulation studies.

*J. P. Lehoczky acknowledges the support of the Air Force Office of Scientific Research at Carnegie-Mellon University, Grant No. AFOSR-74-2642B.

**D. P. Gaver acknowledges the support of the Command and Control Technical Center, Defense Communications Agency.

While we emphasize applications to models of multiprogramming systems in this paper, the method described here may be employed for treating other job-shop-like problems.

1.2 Background

Multiprogramming computer systems have often been represented by some variant of the classical Markovian cyclic queueing, or repairman, model described by Feller (1957). Early work by Gaver (1967) and Rice (1971) has been extended and improved upon by many others, e.g. Baskett et al. (1975), Buzen (1973), Chandy (1975), Reiser and Kobayashi (1974) to name a few. Validation of these models, and applications to planning computer systems, have been reported by Giammo (1976), and Rose (1976). The models have, by and large, depicted jobs or programs as passing in a probabilistic (Markovian) manner from server to server in a closed network of queues forming before the various servers. The latter are identified as one or more Central Processing Units (CPU), and one or more Peripheral Processors (PP), such as disk or tape drives, or input-output devices.

Papers by Jackson (1963) and Gordon and Newell (1967) early pointed out the simple "product form" structure of the long-run (or steady-state, or stationary) joint distribution of jobs present at each server in certain closed cyclic networks.

The latter solution holds if a fixed number of jobs remain in the system, and if job holding or processing times at Server i are identically and independently exponentially distributed, and transitions of jobs from server to server are Markov, each transition from Server i to Server j having the same probability p_{ij}. For other conditions see Chandy (1975). The latter assumptions have frequently been made in modeling computer system behavior. Baskett, et al. have also obtained product-form solutions for situations involving different job types, so that the service time of Job Type k at Server i has rate parameter λ_{ki}, and the between-servers transition probabilities are specified as $p_{ij}^{(k)}$. However, special queue disciplines must be assumed in order to achieve the results. The first-come, first-served (FCFS) discipline is not among these, except when identical service rates prevail. And yet FCFS is more representative of PP behavior than is, for example, a processor-sharing discipline.

Computer system modeling based on direct application of the simplest Jackson-Gordon-Newell results is subject to question on several grounds. First, service or holding times need not be exponentially distributed (nor independent). Second, a mixture of job types may be expected in many real computing systems. For example, some jobs, e.g. those termed compute-bound, may require relatively extensive CPU activity, and spend relatively little time at the PP's; others, the I/O or peripheral-bound, have characteristics that are the reverse. Also, in real

systems different jobs will make heavy demand for information on their own particular PP's, systematically ignoring others. This behavior is, in principle, not well-represented by the homogeneous character of the usual cyclic queueing assumptions. Additionally, non-exponential, e.g. hypo or hyperexponential, service times are of course not properly represented by simple exponentials. One remedy for this deficiency is to utilize phase-type distributions, see Cox and Miller (1965) and also Neuts (1977). One example of a phase type distribution is the gamma or Erlang, thought of as representing a sum of, say, k independently and exponentially distributed random variables. Specification of the phase of service in progress is all that is necessary to render the service process Markov; when the k^{th} phase terminates, service is complete. More generally, phase distributions arise as the hitting or first-passage time of a continuous-time finite Markov chain with one absorbing state; a service terminates when the absorbing state is hit. The class of phase distributions is large, containing distributions exhibiting features such as bimodality and nearly spiked (delta function) behavior. While Markov queueing network analysis to include phase type service is in principle not difficult, in practice, it may result in rather serious computational problems, for the state space necessary to describe the system easily becomes enormous. This is especially true when it becomes necessary to include information concerning the order of jobs, classified as to types, in the state description. While such an "exact" analysis may be carried out for small

Markovian networks, it is advantageous to reduce the state space size in order to facilitate numerical calculations. For that purpose we therefore introduce and evaluate a weighted processor sharing FCFS model.

2. A Numerical Approach Via an Approximating Process

An appropriate Markov central server model of certain multiprogramming computer systems with realistic service protocols can be constructed by specifying an elaborate, large, state space. An example of such an approach is that of Gaver and Humfeld (1976), and Humfeld (1977). In the latter investigations a computational method is devised for deriving the equations of probability balance for a system having two job types and FCFS discipline at one CPU and at each of k different PPs. There are J_i jobs (programs) of each type; $i = 1,2$.

Such a model may, unfortunately, be far too large to be solved numerically for realistic systems. In the next section we introduce a stochastic process, Markov in the number of jobs of each type at each processor, that approximates the behavior of the more complex process required to describe the effect of FCFS disciplines. Apparently a product form analytical solution will not be available for our new approximating process. However, by eliminating reference to both number present and type order, the state space size is dramatically reduced. In terms of the approximating processes one may also more easily study many different queue disciplines as they effect performance in a multitype multiprogramming environment. In effect, processor sharing, FCFS and various priority disciplines may be given a unified treatment.

3. Weighted Processor Sharing FCFS Approximation

For clarity of exposition, we assume that each of the service centers has a FCFS queue discipline and that there are two types of customers. The system contains a fixed number, say J_i, of jobs of type i, i = 1,2. On device j, customers of type i have an exponential service time with parameter μ_{ji} (j = 1 and 2 refer to PP1 and PP2, while j = 0 refers to the CPU). We define $N_{ji}(t)$ to be the number of customers of type i at service center j, i = 1,2; j = 0,1,2. With a processor sharing queue discipline at device j, $\{(N_{j1}(t), N_{j2}(t)), t \geq 0\}$ forms a Markov chain where the probability a customer of type i completes service in $[t,t + dt]$ is given to be $\mu_i N_{ji}(t)\, dt/(N_{j1}(t) + N_{j2}(t)) + o(dt)$, i = 1,2. With a FCFS queue discipline, these probabilities are completely dependent upon the ordering in the queue. They are $\mu_1 dt$ and 0 if a type 1 job is in service, or 0 and $\mu_2 dt$ if a type 2 job is in service. To preserve the Markov property and properly model the system one must keep track of the order of the jobs in the queue. This approach was adopted by Gaver and Humfeld (1976) for exponential service times. The resulting state space has

$$\binom{J_1 + J_2}{J_1} \binom{J_1 + J_2 + k}{k}$$

states if there are k distinct PP's. This number must be greatly increased if service distributions of phase type are

allowed. This approach is thus seriously limited by the size of the state space.

We would like to develop transition probabilities which approximate FCFS behavior but which are based only on $N_{ji}(t)$, $i = 1,2$; $j = 0,1,2$, not the ordering in the queue. Let us focus on a single service center with $N_i(t)$ jobs of type i present at time i and exponential (μ_i) service rates $(i = 1,2)$. In the spirit of processor sharing, we approximate FCFS as follows: let $\Delta_i(t) = N_i(t + dt) - N_i(t)$, $i = 1,2$. We assume

$$P(\Delta_1(t)=-1,\ \Delta_2(t)=0 \mid \underset{\sim}{N}(t)) = \mu_1 \left(\frac{N_1(t)/\mu_1}{N_1(t)/\mu_1 + N_2(t)/\mu_2} \right) dt + o(dt)$$

$$P(\Delta_1(t)=0,\ \Delta_2(t)=-1 \mid \underset{\sim}{N}(t)) = \mu_2 \left(\frac{N_2(t)/\mu_2}{N_1(t)/\mu_1 + N_2(t)/\mu_2} \right) dt + o(dt)$$

$$P(\Delta_1(t)=0,\ \Delta_2(t)=0 \mid \underset{\sim}{N}(t)) = 1 - \frac{N_1(t) + N_2(t)}{N_1(t)/\mu_1 + N_2(t)/\mu_2}\ dt + o(dt)\ . \tag{3.1}$$

The motivation for the suggested approximation is as follows. Over a short period of time $[t, t + dt]$, only the job currently in service has positive probability of completing service. This is in sharp contrast to (3.1) where both types have positive probability of a service completion. Nevertheless, if

we focus on a longer period of time, say the amount of time needed to service all customers currently in the queue, then on the average $N_i(t)/\mu_i$ time units will be spent servicing type i jobs, i = 1,2. Hence the server will spend approximately a fraction $(N_i(t)/\mu_i)/(N_1(t)/\mu_1 + N_2(t)/\mu_2)$ of this total time servicing type i customers. Approximating the system using (3.1) amounts to applying this fraction, which changes dynamically, to every interval of length dt, rather than just to the longer interval in which every customer is served. The approximation is intended to be applied only for steady state calculations, since it does not model the actual dynamics of a FCFS queue discipline faithfully. We note in passing that this is only a first order approximation. It may be improved by applying the fraction $E(X_1/(X_1 + X_2))$ to each dt time interval where X_1 and X_2 are independent gamma random variables with parameters $(N_1(t),\mu_1)$ and $(N_2(t),\mu_2)$. This tends to improve the approximations, but at the cost of a much more complicated analysis. It may, however, be quite important when phase type service distributions are assumed.

Equations (3.1) are a special case of more general transition probabilities for single server queues with multiple customer types. Suppose there are T types and $N_i(t)$ is the number of type i customers $(1 \leq i \leq T)$ enqueued. Let $\Delta_i(t) = N_i(t + dt) - N_i(t)$. Equations (3.1) can be generalized as follows

$$P(\Delta_i(t)=-1,\ \Delta_j(t)=0,\ j \neq i \mid \underset{\sim}{N}(t)) = \frac{\mu_i N_i(t)}{\sum_{j=1}^{T} a_{ij} N_j(t)}\, dt + o(dt)$$

(3.2)

$$P(\Delta_i(t) = 0,\ 1 \leq i \leq T \mid \underset{\sim}{N}(t)) = 1 - \frac{\sum_{i=1}^{T} \mu_i N_i(t)}{\sum_{j=1}^{T} a_{ij} N_j(t)}\, dt + o(dt),$$

Equations (3.2) provide a parametrized family of single server queues. By adjusting the parameters $\{a_{ij},\ 1 \leq i, j \leq T\}$ one can create many different service disciplines. For example, if $a_{ij} = \mu_i/\mu_j$, the resulting discipline is approximately FCFS. The case $a_{ij} = 1$ corresponds to processor sharing. Typically one would choose $a_{ii} = +1$ and $a_{ij} = 1/a_{ji}$. By letting a_{ij} approach 0 or ∞ a priority (of i over j or vice versa) will be created; as $a_{ij} \to 0$, type i jobs are given priority over type j. The coefficients a_{ij} provide relative weights for each type of job at a particular server. See previous work by Lehoczky and Gaver (1977) utilizing a similar approach; that paper also presents numerical validations.

The parameters a_{ij} have intuitive meaning as explained before, and the parametric framework allows for the possibility of optimizing some system measure of effectiveness (such as idleness or queue lengths) by appropriate choice of the a_{ij}. Once an optimal set of a_{ij} has been found, it can be converted into a scheduling algorithm which allocates a particular weight to each type of customer.

This methodology (equations (3.1) or (3.2)) can easily accommodate phase distributions, but at the price of a possibly great increase in the size of the state space. Suppose that at a particular server the service distribution for type i customers i of phase type with n_i+1 states, initial distribution α_i, service or holding time vector $\underset{\sim}{\mu}_i$ and transition probabilities $\underset{\sim}{P}_i$, $1 \leq i \leq T$. We identify each state of each phase distribution as a type of job: there are $S = \sum_{i=1}^{T} n_i$ types. Let $N_{ij}(t)$ be the number of type i jobs in phase j at time t, $1 \leq j \leq n_i$, $1 \leq i \leq T$. Here $\sum_{j=1}^{n_i} N_{ij}(t) = N_i(t)$, the number of jobs of type i.

Coefficients $\{a_{ij}, 1 \leq i, j \leq S\}$ can now be defined by establishing the relationship between types i and j. Equations (3.2) can be easily applied with T replaced by S. While the number of states is potentially enormous, many interesting cases can be handled by solving the steady state balance equations by Gauss-Seidel iteration methods.

We have described the situation of a single service center. The central server model consists of many servers (CPU and PP's). The servers are assumed to be linked in a Markov fashion. When a type i job completes CPU service, it moves to PP_j with probability α_{ij}, $\alpha_{ij} \geq 0$, $\sum_{j=1}^{k} \alpha_{ij} = +1$. Upon completing service at PP_j, it moves back to the CPU. The number of jobs of each type is typically held fixed, but this can be modified to allow them to fluctuate, with the total fixed, in a straightforward fashion. One can also allow the total number of jobs in the system to vary with time although such models are of lesser importance.

4. An Illustration

To illustrate the quality of the approximation given by (3.1) we give an exact analysis of a very simple situation in which closed form expressions can be obtained. A detailed numerical analysis of a more realistically sized system will be given subsequently.

We study the simplest possible multitype central server model with a CPU, two PP's and two job types, each with one job. To further simplify the calculations, type i jobs always move to PPi, i = 1,2 and then back to the CPU. Assume service rates λ_i, i = 1,2 at the CPU and 1 at each PP.

Assuming the CPU is operating in a FCFS fashion, the state space has 5 states defined by the number of each type at the CPU and the type of job currently in service if both types are present. Here x_i will be used to denote the long-run probability that the system inhabits the state having label i.

STATE	LABEL	BALANCE EQUATIONS	
(0,0)	1	$2x_1 = \lambda_1 x_2 + \lambda_2 x_3$	
(1,0)	2	$(\lambda_1 + 1)x_2 = x_1 + \lambda_2 x_5$	
(0,1)	3	$(\lambda_2 + 1)x_3 = x_1 + \lambda_1 x_4$	(4.1)
(1,1,1)	4	$\lambda_1 x_4 = x_2$	
(1,1,2)	5	$\lambda_2 x_5 = x_3$	
		$1 = x_1 + x_2 + x_3 + x_4 + x_5$	

Equations (4.1) can be routinely solved to give

$$
\begin{aligned}
x_1 &= (\lambda_1\lambda_2 + \lambda_1 + \lambda_2)/D \ , \\
x_2 &= (2 + \lambda_2)/D \ , \\
x_3 &= (2 + \lambda_1)/D \ , \\
x_4 &= (2 + \lambda_2)/\lambda_1 D \ , \\
x_5 &= (2 + \lambda_1)/\lambda_2 D_2 , \\
D &= [(\lambda_1 + 1)^2(\lambda_2 + 1)^2 - 1]/\lambda_1\lambda_2
\end{aligned}
\tag{4.2}
$$

The model has only 4 states if the approximation outlined in (3.1) is used. States 4 and 5 in (4.1) merge into a single state; this represents the advertised economy of the proposed approximation. Here y_i denotes the long-run probability of inhabiting label state i.

STATE	LABEL	BALANCE EQUATIONS
(0,0)	1	$2y_1 = \lambda_1 y_2 + \lambda_2 y_3$
(1,0)	2	$(\lambda_1 + 1)y_2 = y_1 + (1/\lambda_1 + 1/\lambda_2)^{-1} y_4$
(0,1)	3	$(\lambda_2 + 1)y_3 = y_1 + y_4(1/\lambda_1 + 1/\lambda_2)^{-1}$
(1,1)	4	$2(1/\lambda_1 + 1/\lambda_2)^{-1} y_4 = y_2 + y_3$
		$1 = y_1 + y_2 + y_3 + y_4$

(4.3)

The solution is easily found to be given by

$$
\begin{aligned}
y_1 &= \lambda_1\lambda_2(2\lambda_1\lambda_2 + \lambda_1 + \lambda_2)/E \\
y_2 &= 2\lambda_1\lambda_2(\lambda_2 + 1)/E \\
y_3 &= 2\lambda_1\lambda_2(\lambda_1 + 1)/E \\
y_4 &= (\lambda_1 + \lambda_2)(2 + \lambda_1 + \lambda_2)/E \\
E &= (\lambda_1\lambda_2 + \lambda_1 + \lambda_2 + 2)(2\lambda_1\lambda_2 + \lambda_1 + \lambda_2)
\end{aligned}
\tag{4.4}
$$

One can compare the results by aggregating x_4 and x_5, so the state spaces are the same. First $x_1 \equiv y_1$, the state which corresponds to CPU idleness. For the other three states the approximation is not exact. We give the percent error of y_i computed by $|x_i - y_i|/y_i$ where x_4 is the aggregation of x_4 and x_5.

$$
\begin{aligned}
|x_2 - y_2|/x_2 &= \lambda_2|\lambda_2 - \lambda_1|/(2 + \lambda_2)(2\lambda_1\lambda_2 + \lambda_1 + \lambda_2)) \\
|x_3 - y_3|/x_3 &= \lambda_1|\lambda_2 - \lambda_1|/(2 + \lambda_1)(2\lambda_1\lambda_2 + \lambda_1 + \lambda_2)) \\
|x_4 - y_4|/x_4 &= \lambda_1\lambda_2(\lambda_2 - \lambda_1)^2/(2\lambda_1\lambda_2 + \lambda_1 + \lambda_2)(\lambda_1^2 + 2\lambda_1 + \lambda_2^2 + 2\lambda_2))
\end{aligned}
\tag{4.5}
$$

One may make a detailed analysis of (4.5). The overall result is that the percentage errors are very small unless an extreme case is chosen. To illustrate we focus on $|x_2 - y_2|/x_2$ and let $\lambda_2 = k\lambda_1$. In this case the percent error is given by

$$\frac{|x_2 - y_2|}{x_2} = \frac{k(k-1)\lambda_1}{(2 + k\lambda_1)(2k\lambda_1 + k + 1)} \qquad (4.6)$$

For fixed k, this expression can be maximized as a function of λ_1 by selecting $\lambda_1 = \sqrt{k+1}/k$. Substituting this value into (4.6) gives

$$\frac{|x_2 - y_2|}{x_2}\text{max} = \frac{k - 1}{(2 + \sqrt{k+1})^2} \qquad (4.7)$$

The following table shows the maximum percent error as a function of k.

k	$(k-1)/(2 + \sqrt{k+1})^2$
1	0%
2	7.1%
3	12.5%
4	16.7%
5	25.2%
10	31.8%
50	59.8%
100	68.9%
∞	100%

One can see that except in very extreme cases, the percent error will be very small indeed. A similar analysis can be carried out on the other terms with similar results.

5. A Second Approximation

Another approximation method can be used which offers several advantages, but some disadvantages, over the weighted processor sharing example. We illustrate this for a single service center. Rather than keep track only of the number of each job type at the service center as in the first approximation, we expand the state description to include the number of each type awaiting service and the identity of the customer in service. Arrivals to the queue and departures from the queue are easily handled since the identity of the customer in service and the number at the center is known. An approximation is required to identify the type of the next customer to enter service when a customer leaves, because the order within the queue is not known. We adopt a random service order policy approximation. Specifically, if $W_i(t)$ represents the number of type i jobs waiting for service, i = 1,2 and $I(t) = i$ if a type i job is in service, then we assume, letting $\Delta_i = W_i(t + dt) - W_i(t)$

$$P(\Delta_1=-1, \Delta_2=0, I(t+h)=1 \mid \underset{\sim}{W}(t), I(t)=1) = \mu_1 \frac{W_1(t)}{W_1(t)+W_2(t)} dt + o(dt)$$

$$P(\Delta_1=0, \Delta_2=-1, I(t+h)=2 \mid \underset{\sim}{W}(t), I(t)=1) = \mu_1 \frac{W_2(t)}{W_1(t)+W_2(t)} dt + o(dt)$$

$$P(\Delta_1=-1, \Delta_2=0, I(t+h)=1 \mid \underset{\sim}{W}(t), I(t)=2) = \mu_2 \frac{W_1(t)}{W_1(t)+W_2(t)} dt + o(dt)$$

$$P(\Delta_1=0, \Delta_2=-1, I(t+h)=2 \mid \underset{\sim}{W}(t), I(t)=2) = \mu_2 \frac{W_2(t)}{W_1(t)+W_2(t)} dt + o(dt) . \quad (5.1)$$

This can be easily extended to many service centers and provides a second approximation to FCFS service disciplines. The numerical accuracy will be assessed in the next section.

This approximation requires a slightly larger state space than the weighted processor sharing (3.1) approximation for exponential service distributions. Nevertheless if phase type distributions are introduced the state space for (5.1) is much smaller than for (3.1). Phase type distributions can be very easily accommodated. Another advantage is that (5.1) provides better accuracy than (3.1), although both give excellent results. On the negative side (5.1) cannot be conveniently changed to allow for other types of queue disciplines.

6. Numerical Illustrations

Numerical studies were carried out using (3.1) and (5.1) and were compared with the results of Gaver and Humfeld (1976). Below we present the results for three cases. In all cases there are two PP's and two job types. The CPU service rates are λ_1, λ_2, while (μ_{11}, μ_{12}) are those for PP1 and (μ_{21}, μ_{22}) for PP2. α_{11} and α_{21} are the probabilities a type 1 or type 2 job moves to PP1 upon leaving the CPU. J_i is the fixed number of type i jobs, i = 1,2.

Method 1 refers to the results of Gaver and Humfeld (1976) obtained by a Gauss-Seidel iteration of the balance equations for the exact system. Method 2 refers to the steady state results obtained using a Gauss-Seidel iteration of the balance equations derived from (3.1), the weighted processor sharing approximation. Method 3 refers to the approximation based on a random source order (5.1). The results were obtained using simulation. The results are based on 120,000 system transitions, and estimated standard errors are provided with each quantity.

The numerical values given in the previous tables illustrate the exceptional accuracy of both approximation method across a broad number of cases. The largest absolute error is .02, thus the approximations are extremely useful. The tables illustrate that method 3 (based on (5.1)) is more accurate than method 2 (based on (3.1)). Method 2 allocates slightly too much time to slow jobs, slightly too much to fast jobs. This results in the slight but

systematic differences between methods 1 and 2. In view of the minor differences, the complicated correction mentioned in section 3 is certainly not called for.

The tables illustrate that method 3 should be used whenever possible. This will be especially true if phase type distributions are required. The accuracy is better than method 2 and the state space requirements may be substantially reduced. Method 2 can, however, serve as a starting place for calculating approximate closed form expressions for occupancies and idleness probabilities, say by using a diffusion approximation analysis.

ase I

$\lambda_1 = 15$ $\lambda_2 = 25$ $\alpha_{11} = .75$

$\mu_{11} = 10$ $\mu_{12} = 20$ $\alpha_{21} = .25$

$\mu_{21} = 20$ $\mu_{22} = 10$

		Occupancy			Idleness		
(J_1,J_2)	Method	CPU	PP1	PP2	CPU	PP1	PP2
(1,1)	1	.830	.582	.588	.406	.519	.501
	2	.827	.566	.607	.403	.524	.483
	3	.830(.003)	.587(.004)	.587(.006)	.409(.001)	.518(.002)	.505(.005)
(1,2)	1	1.172	.613	1.215	.336	.537	.284
	2	1.163	.590	1.247	.334	.544	.266
	3	1.172(.013)	.616(.003)	1.219(.013)	.337(.007)	.535(.004)	.283(.001)
(2,1)	1	1.342	1.093	.565	.282	.341	.556
	2	1.343	1.074	.582	.277	.345	.538
	3	1.346(.012)	1.085(.019)	.576(.007)	.281(.006)	.342(.005)	.533(.005)
(1,3)	1	1.444	.613	1.943	.305	.558	.165
	2	1.425	.587	1.988	.306	.565	.151
	3	1.444(.019)	.627(.006)	1.940(.018)	.309(.006)	.556(.004)	.170(.003)
(2,2)	1	1.779	1.106	1.115	.224	.367	.355
	2	1.777	1.077	1.146	.220	.372	.334
	3	1.794(.024)	1.104(.010)	1.112(.015)	.221(.003)	.369(.005)	.359(.005)
(3,1)	1	1.839	1.626	.535	.218	.245	.594
	2	1.844	1.608	.547	.214	.248	.579
	3	1.826(.009)	1.637(.009)	.547(.009)	.221(.002)	.242(.002)	.591(.003)
(3,3)	1	2.847	1.572	1.581	.138	.296	.286
	2	2.846	1.536	1.618	.134	.300	.268
	3	2.836(.076)	1.600(.036)	1.583(.042)	.140(.009)	.297(.006)	.292(.008)
(2,5)	1	2.672	.998	3.330	.182	.441	.096
	2	2.619	.959	3.422	.182	.447	.085
	3	2.684(.048)	1.021(.022)	3.315(.047)	.180(.006)	.438(.003)	.100(.006)
(5,2)	1	3.459	2.690	.851	.115	.169	.491
	2	3.475	2.664	.861	.111	.172	.477
	3	3.456(.078)	2.709(.077)	.854(.025)	.118(.005)	.172(.008)	.493(.014)

Case II

$\lambda_1 = 0,526$ $\lambda_2 = 0.339$ $\alpha_{11} = 1.0$

$\mu_{11} = 1.0$ $\mu_{12} = 1.0$ $\alpha_{21} = 0.0$

$\mu_{21} = 1.0$ $\mu_{22} = 1.0$

		Occupancy			Idleness		
(J_1,J_2)	Method	CPU	PP1	PP2	CPU	PP1	PP2
(1,1)	1	1.610	.200	.190	.059	.800	.810
	2	1.608	.205	.187	.059	.795	.813
	3	1.609(.005)	.201(.002)	.192(.003)	.060(.002)	.800(.002)	.809(.00
(1,2)	1	2.575	.135	.290	.014	.865	.753
	2	2.571	.136	.292	.014	.864	.754
	3	2.574(.001)	.135(.003)	.293(.002)	.014(.001)	.866(.003)	.751(.00
(2,1)	1	2.500	.353	.148	.018	.713	.852
	2	2.503	.351	.146	.018	.709	.854
	3	2.503(.011)	.351(.010)	.149(.002)	.018(.001)	.715(.006)	.852(.00

Case III

$\lambda_1 = 0.6$ $\lambda_2 = 0.8$ $\alpha_{11} = .55$

$\mu_{11} = 1.2$ $\mu_{12} = 1.0$ $\alpha_{21} = .30$

$\mu_{21} = 0.9$ $\mu_{22} = 0.5$

		Occupancy			Idleness		
(J_1,J_2)	Method	CPU	PP1	PP2	CPU	PP1	PP2
(4,4)	1	6.107	.345	1.548	.011	.742	.372
	2	6.152	.347	1.501	.009	.742	.376
	3	6.113(.062)	.342(.015)	1.561(.060)	.012(.003)	.744(.008)	.374(.00

REFERENCES

Baskett, F., Chandy, K.M., Muntz, R.R., and Palacios-Gomez, F. (1975). "Open, closed and mixed networks of queues with different classes of customers," JACM 22, pp. 248-260.

Buzen, J. (1973). "Computational algorithms for closed queueing networks with exponential servers," Comm. ACM 16, pp. 527-531.

Chandy, K. M., Herzog, U. and Woo, L. (1975). "Approximate analysis of general queueing networks," IBM J. Res. Devel. 19, 1, pp. 43-49.

Cox, D.R. and Miller, H. D. (1965). The Theory of Stochastic Processes, John Wiley and Sons, New York, N.Y.

Feller, W. (1957). An Introduction to Probability Theory and Its Applications, Vol. I, John Wiley and Sons, New York, N.Y.

Gaver, D.P. (1967). "Probability models for multiprogramming computer systems," JACM, 14, pp. 423-438.

Gaver, D. P. and Humfeld, G. (1976). "Multitype multiprogramming models," Acta Informatica 7, pp. 111-121.

Gaver, D. P. and Lehoczky, J. P. (1977). "Models for work backlogs at computers that time-share heterogeneous users." Naval Postgraduate School Tech. Report (in preparation).

Giammo, T. (1976). "Validation of a computer performance model of the exponential queueing network family," Proc. of the International Symposium on Computer Performance Modeling, Measurement and Evaluation, Harvard University, Cambridge, Mass., pp. 44-58.

Gordon, W. and Newell, G. F. (1967). "Closed queueing systems with exponential servers," Ops. Res. 15, pp. 254-265.

Humfeld, G. (1977). "Numerical methods for analyzing multitype multiprogramming computer system queues." Ph.D. Thesis, Department of Operations Research, Naval Postgraduate School, Monterey, California.

Jackson, J. R. (1963). "Job shop-like queueing systems," Management Science, 10, pp. 131-142.

Neuts, M. (1977). "Algorithms for the waiting-time distributions, under various queue disciplines, in the M/G/1 queue with service time distributions of phase type." Paper in Algorithmic Methods in Probability, TIMS-North Holland Studies in Management Science (to appear).

Reiser, M. and Kobayashi, H. (1974). "Accuracy of the diffusion approximation for some queueing systems," IBM J of Res. Devel. 18, pp. 110-124.

Rice, D. R. (1971). "An analytical model for computer system performance analysis," Ph.D. Dissertation, University of Florida.

Rose, C. (1976). "Validation of a queueing model with classes of customers," Proc. International Symposium on Computer Performance Modeling, Measurement and Evaluation, Harvard University, Cambridge, Mass., pp. 318-325.